Racing with Deep Reinforecment Learning

Walter A. Freire, Kevin M. Freire
andrei.freire@ryerson.ca, kfreirea@ryerson.ca
Ryerson University, Toronto, Canada.

***Abstract*—The group members are Walter Freire and Kevin Freire and the project topic we will be working on is autonomous driving using reinforcement learning.**

I. Intorduction

Reinforcement Learning (RL) has been used to accomplish various robotic tasks such as manipulation, navigation, flight, interacion, motion planning and many more. When implementing an RL model into the real world, due to its high sample complexity, safety requirements and cost it is common to train the RL agent in a simulation. In robotic RL many expertise are required, such as access to physical robot, an accurate robot model for simulation, a diverse training mechanism and customizability of the training procedure such as modifying the neural network and the loss function [1]. However, all this is available and simulation problems are covered with DeepRacer.

DeepRacer supports state-of-the-art deep RL algorithms [29], simulaitons with OpenAI Gym interface [17], distributed rollouts and integration with could services. DeepRacer uses a 1/18th scale car of a physical robot that uses RL for navigation of race track with a fisheye lens camera [1] with other specs such as GPU, live streaming view and more. In this paper the robot model in DeepRacers simulation is used along with the multiple provided race tracks. With the provided sources the RL modelpolicy is trainined with diffirent simulation parameters and multiple tracks in parallel using distributed rollouts.

DeepRacer is a great platform for learning an end-to-end policy for navigating througha race track, that uses a single grayscale camera image as observation and discretized throttle/steering ad actions. The training is done in simulation using Proximal Policy Optimization (PPO) algorithm [31], which can converge in under 5 minutes and about 5000 simulation steps.

The purpose of slecting this platform and training a vehicle with RL is because vehicles are normally trained using deep neural networks with large datasets of videos that require high comutational power. Deepracer provides an interactive experience in training a vehicle to drive itself through a race track with minimal supervision. It also provides a jail-broken version in order to hack into their source code to implement our own network architecture, customize the action-space, reward functions and much more. In addition to learning about the Deepracers algorithms, there was alot of hands on experience using AWS, and creating Jupyter notebooks for building a docker and SageMaker Images and for simulation. The rest of the article will focus on the Problem statements and it's Environment/Dataset, Methods and Model, last is Results and Discussion.

II. Problem Statement and Environement/Dataset

Define the problem more rigorously, perhaps with some examples. Describe the environment or the dataset (e.g., where did you get the environment or the data from?, What is the reward? What are the states? What are the actions? etc.)

III. Methods and Model

In this section, you should describe your solutions to the problem you described in previous sections. What preprocessing and/or feature extraction methods did you consider and apply. did you make any changes/corrections to the reward? Why? What reinforcement learning (RL) methods or models did you consider for solving this problem? What are the performance metrics that you considered for evaluating, comparing, or choosing the best model/agents? Why did you choose these models, methods, preprocessing, and evaluation metrics?

IV. Results and Discussion

Discuss the most important results and observations. Specifically, I would like to see some analysis on what you think are the failures of the first set of results that you obtained. This can be failures in terms of attaining good performance, failing in specific conditions, or the algorithm being sample inefficient (i.e., you need a lot of episodes to train). How did you try to mitigate these issues? Did your approach work? Also, present some tables showing your progress and some figures and graphs highlighting your most important observations and results. Please explain if you were to continue the project, what you will do next.

V. Implementation and Code

Briefly describe what python packages, pieces of code from GitHub, blogs, or other sources (e.g., research papers) did you use to implement your solution.

VI. References

Please make sure you cite any sources you have used to complete the project

[1] DeepRacer Article [29] Deep racer reference 29 [17] Deep racer reference 17 [31] Deep racer reference 31